LOG4J - ARCHITECTURE

Log4j API has been designed in layered where each layer provides different object which performs different tasks. This makes design flexible and very much extendable in future based on need.

There are two type of objects available with Log4j framework.

- **Core Objects:** These are mandatory objects of the framework and required to use the framework.
- **Support Objects:** These are optional objects of the framework and support core objects to perform addition but important tasks.

Core Objects:

Logger Object:

The top level layer is Logger which provides Logger object. The Logger object is responsible for capturing logging information and they are stored in a namespace hierarchy.

Layout Object:

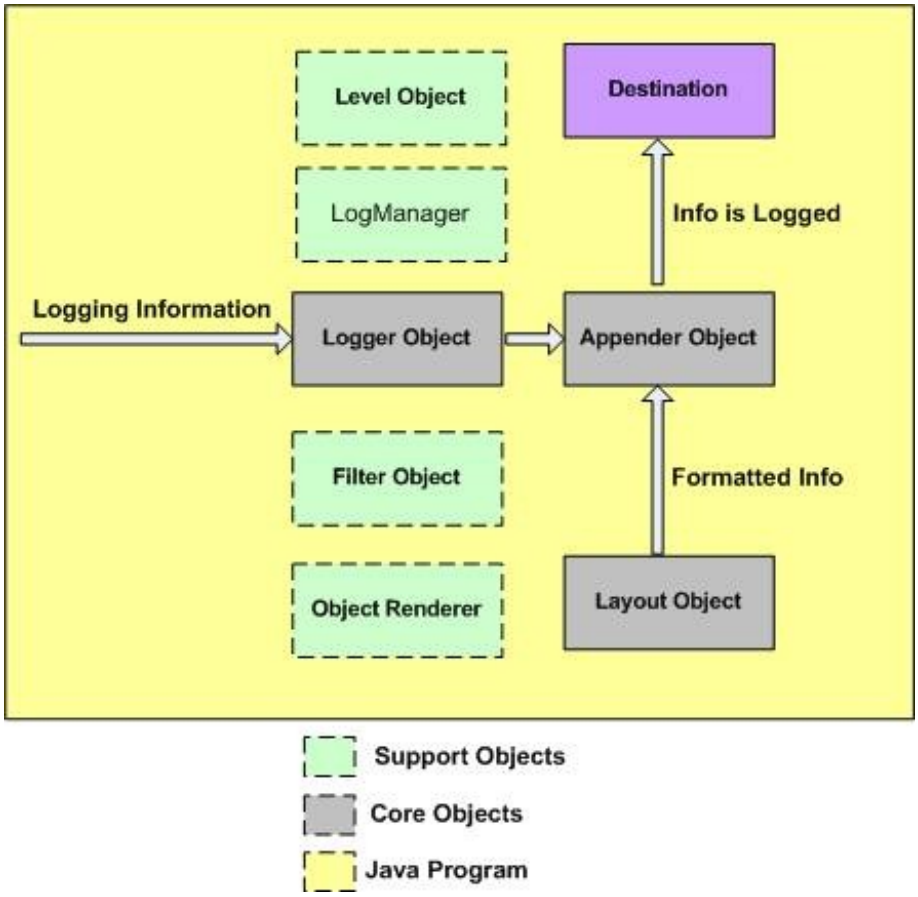
The layer provides objects which are used to format logging information in different styles. Layout layer provides support to appender objects to before publishing logging information.

Layout objects play an important role in publishing logging information in a way that is human-readable and reusable.

Appender Object:

This is lower level layer which provides Appender object. The Appender object is responsible for publishing logging information to various preferred destinations such as a database, file, console, UNIX Syslog etc.

Following is a virtual diagram showing different components of Log4J Framework:



Support Objects:

There are other important objects in the log4j framework that play a vital role in the logging framework:

Level Object:

The Level object defines the granularity and priority of any logging information. There are seven levels of logging defined within the API: OFF, DEBUG, INFO, ERROR, WARN, FATAL, and ALL.

Filter Object:

The Filter object is used to analyze logging information and make further decisions on whether that information should be logged or not.

An Appender objects can have several Filter objects associated with them. If logging information is passed to a particular Appender object, all the Filter objects associated with that Appender need to approve the logging information before it can be published to the attached destination.

ObjectRenderer:

The ObjectRenderer object is specialized in providing a String representation of different objects passed to the logging framework. This object is used by Layout objects to prepare the final logging information.

LogManager:

The LogManager object manages the logging framework. It is responsible for reading the initial configuration parameters from a system-wide configuration file or a configuration class.